Markovian Testing Equivalence and Exponentially Timed Internal Actions

Marco Bernardo
Università di Urbino "Carlo Bo" – Italy
Istituto di Scienze e Tecnologie dell'Informazione

In the theory of testing for Markovian processes developed so far, exponentially timed internal actions are not admitted within processes. When present, these actions cannot be abstracted away, because their execution takes a nonzero amount of time and hence can be observed. On the other hand, they must be carefully taken into account, in order not to equate processes that are distinguishable from a timing viewpoint. In this paper, we recast the definition of Markovian testing equivalence in the framework of a Markovian process calculus including exponentially timed internal actions. Then, we show that the resulting behavioral equivalence is a congruence, has a sound and complete axiomatization, has a modal logic characterization, and can be decided in polynomial time.

1 Introduction

Markovian behavioral equivalences are a means to relate and manipulate formal models with an underlying continuous-time Markov chain (CTMC) semantics. Various proposals have appeared in the literature, which are extensions of the traditional approaches to the definition of behavioral equivalences. Markovian bisimilarity [14, 13, 5] considers two processes to be equivalent whenever they are able to mimic each other's functional and performance behavior stepwise. Markovian testing equivalence [2] considers two processes to be equivalent whenever an external observer is not able to distinguish between them from a functional or performance viewpoint by interacting with them by means of tests and comparing their reactions. Markovian trace equivalence [19] considers two processes to be equivalent whenever they are able to perform computations with the same functional and performance characteristics.

The three Markovian behavioral equivalences mentioned above have different discriminating powers as a consequence of their different definitions. However, they are all meaningful not only from a functional standpoint [17, 11, 7], but also from a performance standpoint. In fact, Markovian bisimilarity is known to be in agreement with an exact CTMC-level aggregation called ordinary lumpability [14, 8], while Markovian testing and trace equivalences are known to be consistent with a coarser exact CTMC-level aggregation called T-lumpability [2, 3].

In this paper, we focus on the treatment of internal actions – denoted by τ as usual – that are exponentially timed. Unlike internal actions of nondeterministic processes, exponentially timed internal actions cannot be abstracted away, because their execution takes a nonzero amount of time and hence can be observed. To be precise, in [14, 6, 1] the issue of abstracting from them has been addressed, but it remains unclear whether and to what extent abstraction is possible, especially if we want to end up with a weak Markovian behavioral equivalence that induces a nontrivial, exact CTMC-level aggregation.

The definition of Markovian bisimilarity smoothly includes exponentially timed internal actions, by applying to them the same exit rate equality check that is applied to exponentially timed visible actions. Unfortunately, this is not the case with Markovian testing and trace equivalences as witnessed by the theory developed for them, which does not admit exponentially timed internal actions within processes.

S. Andova et.al. (Eds.): Workshop on Quantitative Formal Methods:
Theory and Applications (QFM'09)
EPTCS 13, 2009, pp. 13–25, doi:10.4204/EPTCS.13.2

When present, these actions must be carefully taken into account in order not to equate processes that are distinguishable from a timing viewpoint. As an example, given $\lambda, \mu \in \mathbb{R}_{>0}$, processes "$<\tau, \lambda>.\underline{0}$" – which can only execute an exponentially timed internal action whose average duration is $1/\lambda$ – and "$<\tau, \mu>.\underline{0}$" – which can only execute an exponentially timed internal action whose average duration is $1/\mu$ – should not be considered equivalent if $\lambda > \mu$, as the durations of their actions are sampled from different exponential probability distributions. Moreover, if they were considered equivalent, then congruence with respect to alternative and parallel composition would not hold.

With the definition of Markovian testing equivalence given in [2] – which compares the probabilities of passing the same test within the same average time upper bound – there is no way to distinguish between the two processes above, as they pass with probability 1 the test comprising only the success state and with probability 0 any other test, independent of the fixed average time upper bound. In this paper, we show that a simple way to distinguish between the two processes above consists of imposing an additional constraint on the length of the successful computations to take into account.

For instance, if we take a test comprising only the success state, the two processes above pass the test with probability 1 for every average time upper bound if we restrict ourselves to successful computations of length 0. However, if we move to successful computations of length 1 and we use $1/\lambda$ as average time upper bound, it turns out that $<\tau, \lambda>.\underline{0}$ reaches success with probability 1 – as it has enough time on average to perform its only action – whereas $<\tau, \mu>.\underline{0}$ does not – as it has not enough time on average to perform its only action by the deadline. A similar idea applies to Markovian trace equivalence.

After introducing a Markovian process calculus that includes exponentially timed internal actions (Sect. 2), we present a new definition of Markovian testing equivalence that embodies the idea illustrated above (Sect. 3). Then, we show that (i) it coincides with the equivalence defined in [2] when exponentially timed internal actions are absent, (ii) its discriminating power does not change if we introduce exponentially timed internal actions within tests, and (iii) it inherits the fully abstract characterization studied in [2] (Sect. 4). Furthemore, we show that it is a congruence with respect to typical dynamic and static operators (Sect. 5) and has a sound and complete axiomatization for nonrecursive processes (Sect. 6), thus overcoming the limitation to dynamic operators of analogous results contained in [2]. Finally, we show that it has a modal logic characterization (Sect. 7), which is based on the same modal language as [4], and that it can be decided in polynomial time (Sect. 8).

2 Markovian Process Calculus

In this section, we present a process calculus in which every action has associated with it a rate that uniquely identifies its exponentially distributed duration. The definition of the syntax and of the semantics for the resulting Markovian process calculus – MPC for short – is followed by the introduction of some notations related to process terms and their computations that will be used in the rest of the paper.

2.1 Durational Actions and Behavioral Operators

In MPC, an exponentially timed action is represented as a pair $<a, \lambda>$. The first element, a, is the name of the action, which is τ in the case that the action is internal, otherwise it belongs to a set $Name_{\mathrm{v}}$ of visible action names. The second element, $\lambda \in \mathbb{R}_{>0}$, is the rate of the exponentially distributed random variable RV quantifying the duration of the action, i.e., $\Pr\{RV \leq t\} = 1 - \mathrm{e}^{-\lambda \cdot t}$ for $t \in \mathbb{R}_{>0}$. The average duration of the action is equal to the reciprocal of its rate, i.e., $1/\lambda$. If several exponentially timed actions are enabled, the race policy is adopted: the action that is executed is the fastest one.

The sojourn time associated with a process term P is thus the minimum of the random variables quantifying the durations of the exponentially timed actions enabled by P. Since the minimum of several exponentially distributed random variables is exponentially distributed and its rate is the sum of the rates of the original variables, the sojourn time associated with P is exponentially distributed with rate equal to the sum of the rates of the actions enabled by P. Therefore, the average sojourn time associated with P is the reciprocal of the sum of the rates of the actions it enables. The probability of executing one of those actions is given by the action rate divided by the sum of the rates of all the considered actions.

Passive actions of the form $<a, *_w>$ are also included in MPC, where $w \in \mathbb{R}_{>0}$ is the weight of the action. The duration of a passive action is undefined. When several passive actions are enabled, the reactive preselection policy is adopted. This means that, within every set of enabled passive actions having the same name, each such action is given an execution probability equal to the action weight divided by the sum of the weights of all the actions in the set. Instead, the choice among passive actions having different names is nondeterministic. Likewise, the choice between a passive action and an exponentially timed action is nondeterministic.

MPC comprises a CSP-like parallel composition operator [7] relying on an asymmetric synchronization discipline [5], according to which an exponentially timed action can synchronize only with a passive action having the same name. In other words, the synchronization between two exponentially timed actions is forbidden. Following the terminology of [12], the adopted synchronization discipline mixes generative and reactive probabilistic aspects. Firstly, among all the enabled exponentially timed actions, the proposal of an action name is generated after a selection based on the rates of those actions. Secondly, the enabled passive actions that have the same name as the proposed one react by means of a selection based on their weights. Thirdly, the exponentially timed action winning the generative selection and the passive action winning the reactive selection synchronize with each other. The rate of the synchronization is given by the rate of the selected exponentially timed action multiplied by the execution probability of the selected passive action, thus complying with the bounded capacity assumption [14].

We denote by $Act = Name \times Rate$ the set of actions of MPC, where $Name = Name_{\rm v} \cup \{\tau\}$ is the set of action names – ranged over by a, b – and $Rate = \mathbb{R}_{>0} \cup \{*_w \mid w \in \mathbb{R}_{>0}\}$ is the set of action rates – ranged over by $\tilde{\lambda}, \tilde{\mu}$. We then denote by *Relab* a set of relabeling functions $\varphi : Name \rightarrow Name$ that preserve action visibility, i.e., such that $\varphi^{-1}(\tau) = \{\tau\}$. Finally, we denote by *Var* a set of process variables ranged over by X, Y.

Definition 2.1 The set of process terms of the process language $\mathscr{PL}$ is generated by the following syntax:

P	$::=$	$\underline{0}$	inactive process
	$\mid$	$<a, \lambda>.P$	exponentially timed action prefix
	$\mid$	$<a, *_w>.P$	passive action prefix
	$\mid$	$P + P$	alternative composition
	$\mid$	$P \parallel_S P$	parallel composition
	$\mid$	P / H	hiding
	$\mid$	$P[\varphi]$	relabeling
	$\mid$	X	process variable
	$\mid$	$rec\, X : P$	recursion

where $a \in Name$, $\lambda, w \in \mathbb{R}_{>0}$, $S, H \subseteq Name_{\rm v}$, $\varphi \in Relab$, and $X \in Var$. We denote by $\mathbb{P}$ the set of closed and guarded process terms of $\mathscr{PL}$. ■

2.2 Operational Semantics

The semantics for MPC can be defined in the usual operational style, with an important difference with respect to the nondeterministic case. A process term like $<a,\lambda>.\underline{0} + <a,\lambda>.\underline{0}$ is not the same as $<a,\lambda>.\underline{0}$, because the average sojourn time associated with the latter, i.e., $1/\lambda$, is twice the average sojourn time associated with the former, i.e., $1/(\lambda+\lambda)$. In order to assign distinct semantic models to terms like the two considered above, we have to take into account the multiplicity of each transition, intended as the number of different proofs for the transition derivation. The semantic model $[\![P]\!]$ for a process term $P \in \mathbb{P}$ is thus a labeled multitransition system, whose multitransition relation is contained in the smallest multiset of elements of $\mathbb{P} \times Act \times \mathbb{P}$ satisfying the operational semantic rules of Table 1 ($\{\_ \hookrightarrow \_\}$ denotes syntactical replacement; $\{\!|\, , |\!\}$ are multiset parentheses).

We observe that exponential distributions fit well with the interleaving view of parallel composition. Due to their memoryless property, the execution of an exponentially timed action can be thought of as being started in the last state in which the action is enabled. Due to their infinite support, the probability that two concurrent exponentially timed actions terminate simultaneously is zero.

The CTMC underlying a process term $P \in \mathbb{P}$ can be derived from $[\![P]\!]$ iff this labeled multitransition system has no passive transitions, in which case we say that P is performance closed. We denote by $\mathbb{P}_{\mathrm{pc}}$ the set of performance closed process terms of $\mathbb{P}$.

2.3 Exit Rates of Process Terms

The exit rate of a process term $P \in \mathbb{P}$ is the rate at which P can execute actions of a certain name $a \in Name$ that lead to a certain destination $D \subseteq \mathbb{P}$ and is given by the sum of the rates of those actions due to the race policy. We consider a two-level definition of exit rate, with level 0 corresponding to exponentially timed actions and level -1 corresponding to passive actions:

$$\boxed{rate_{\mathrm{e}}(P,a,l,D) \; = \; \begin{cases} \sum\{\!|\, \lambda \in \mathbb{R}_{>0} \mid \exists P' \in D.\, P \xrightarrow{a,\lambda} P' \,|\!\} & \text{if } l = 0 \\ \sum\{\!|\, w \in \mathbb{R}_{>0} \mid \exists P' \in D.\, P \xrightarrow{a,*_w} P' \,|\!\} & \text{if } l = -1 \end{cases}}$$

where each summation is taken to be zero whenever its multiset is empty.

By summing up the rates of all the actions of a certain level l that P can execute, we obtain the total exit rate of P at level l:

$$\boxed{rate_{\mathrm{t}}(P,l) \; = \sum_{a \in Name} rate_{\mathrm{o}}(P,a,l)}$$

where:

$$\boxed{rate_{\mathrm{o}}(P,a,l) \; = \; rate_{\mathrm{e}}(P,a,l,\mathbb{P})}$$

is the overall exit rate of P with respect to a at level l.

If P is performance closed, then $rate_{\mathrm{t}}(P,0)$ coincides with the reciprocal of the average sojourn time associated with P. Instead, $rate_{\mathrm{o}}(P,a,-1)$ coincides with $weight(P,a)$.

2.4 Probability and Duration of Computations

A computation of a process term $P \in \mathbb{P}$ is a sequence of transitions that can be executed starting from P. The length of a computation is given by the number of transitions occurring in it. We denote by $\mathscr{C}_{\mathrm{f}}(P)$ the multiset of finite-length computations of P. We say that two distinct computations are independent of each other if neither is a proper prefix of the other one. In the following, we concentrate on finite

$$(\text{PRE}_1)\quad \frac{}{<a,\lambda>.P \xrightarrow{a,\lambda} P} \qquad\qquad (\text{PRE}_2)\quad \frac{}{<a,*_w>.P \xrightarrow{a,*_w} P}$$

$$(\text{ALT}_1)\quad \frac{P_1 \xrightarrow{a,\tilde{\lambda}} P'}{P_1 + P_2 \xrightarrow{a,\tilde{\lambda}} P'} \qquad\qquad (\text{ALT}_2)\quad \frac{P_2 \xrightarrow{a,\tilde{\lambda}} P'}{P_1 + P_2 \xrightarrow{a,\tilde{\lambda}} P'}$$

$$(\text{PAR}_1)\quad \frac{P_1 \xrightarrow{a,\tilde{\lambda}} P_1' \qquad a \notin S}{P_1 \,\|_S\, P_2 \xrightarrow{a,\tilde{\lambda}} P_1' \,\|_S\, P_2} \qquad\qquad (\text{PAR}_2)\quad \frac{P_2 \xrightarrow{a,\tilde{\lambda}} P_2' \qquad a \notin S}{P_1 \,\|_S\, P_2 \xrightarrow{a,\tilde{\lambda}} P_1 \,\|_S\, P_2'}$$

$$(\text{SYN}_1)\quad \frac{P_1 \xrightarrow{a,\lambda} P_1' \qquad P_2 \xrightarrow{a,*_w} P_2' \qquad a \in S}{P_1 \,\|_S\, P_2 \xrightarrow{a,\lambda\cdot\frac{w}{weight(P_2,a)}} P_1' \,\|_S\, P_2'}$$

$$(\text{SYN}_2)\quad \frac{P_1 \xrightarrow{a,*_w} P_1' \qquad P_2 \xrightarrow{a,\lambda} P_2' \qquad a \in S}{P_1 \,\|_S\, P_2 \xrightarrow{a,\lambda\cdot\frac{w}{weight(P_1,a)}} P_1' \,\|_S\, P_2'}$$

$$(\text{SYN}_3)\quad \frac{P_1 \xrightarrow{a,*_{w_1}} P_1' \qquad P_2 \xrightarrow{a,*_{w_2}} P_2' \qquad a \in S}{P_1 \,\|_S\, P_2 \xrightarrow{a,*_{norm(w_1,w_2,a,P_1,P_2)}} P_1' \,\|_S\, P_2'}$$

$$(\text{HID}_1)\quad \frac{P \xrightarrow{a,\tilde{\lambda}} P' \qquad a \in H}{P/H \xrightarrow{\tau,\tilde{\lambda}} P'/H} \qquad\qquad (\text{HID}_2)\quad \frac{P \xrightarrow{a,\tilde{\lambda}} P' \qquad a \notin H}{P/H \xrightarrow{a,\tilde{\lambda}} P'/H}$$

$$(\text{REL})\quad \frac{P \xrightarrow{a,\tilde{\lambda}} P'}{P[\varphi] \xrightarrow{\varphi(a),\tilde{\lambda}} P'[\varphi]}$$

$$(\text{REC})\quad \frac{P\{rec X : P \hookrightarrow X\} \xrightarrow{a,\tilde{\lambda}} P'}{rec X : P \xrightarrow{a,\tilde{\lambda}} P'}$$

$$weight(P,a) \;=\; \sum\{\!|\, w \in \mathbb{R}_{>0} \mid \exists P' \in \mathbb{P}.\, P \xrightarrow{a,*_w} P' \,|\!\}$$

$$norm(w_1,w_2,a,P_1,P_2) \;=\; \frac{w_1}{weight(P_1,a)} \cdot \frac{w_2}{weight(P_2,a)} \cdot (weight(P_1,a) + weight(P_2,a))$$

Table 1: Operational semantic rules for MPC

multisets of independent, finite-length computations. Below we define the probability and the duration of a computation $c \in \mathscr{C}_{\mathrm{f}}(P)$ for $P \in \mathbb{P}_{\mathrm{pc}}$, using $\_ \circ \_$ for sequence concatenation and $|\_|$ for sequence length.

The probability of executing c is the product of the execution probabilities of the transitions of c:

$$\boxed{prob(c) \; = \; \begin{cases} 1 & \text{if } |c| = 0 \\ \frac{\lambda}{rate_{\mathrm{t}}(P,0)} \cdot prob(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}}$$

We also define the probability of executing a computation in $C \subseteq \mathscr{C}_{\mathrm{f}}(P)$ as:

$$\boxed{prob(C) \; = \; \sum_{c \in C} prob(c)}$$

whenever C is finite and all of its computations are independent of each other.

The stepwise average duration of c is the sequence of average sojourn times in the states traversed by c:

$$\boxed{time_{\mathrm{a}}(c) \; = \; \begin{cases} \varepsilon & \text{if } |c| = 0 \\ \frac{1}{rate_{\mathrm{t}}(P,0)} \circ time_{\mathrm{a}}(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}}$$

where ε is the empty stepwise average duration. We also define the multiset of computations in $C \subseteq \mathscr{C}_{\mathrm{f}}(P)$ whose stepwise average duration is not greater than $\theta \in (\mathbb{R}_{>0})^*$ as:

$$\boxed{C_{\leq \theta} \; = \; \{\!| \, c \in C \mid |c| \leq |\theta| \wedge \forall i = 1, \dots, |c|. \, time_{\mathrm{a}}(c)[i] \leq \theta[i] \, |\!\}}$$

Moreover, we denote by C^l the multiset of computations in $C \subseteq \mathscr{C}_{\mathrm{f}}(P)$ whose length is equal to $l \in \mathbb{N}$.

We conclude by observing that the average duration of a finite-length computation has been defined as the sequence of average sojourn times in the states traversed by the computation. The same quantity could have been defined as the sum of the same basic ingredients, but this would not have been appropriate as explained in [19, 2].

3 Redefining Markovian Testing Equivalence

The basic idea behind testing equivalence is to infer information about the behavior of process terms by interacting with them by means of tests and comparing their reactions. In a Markovian setting, we are not only interested in verifying whether tests are passed or not, but also in measuring the probability with which they are passed and the time taken to pass them. Therefore, we have to restrict ourselves to $\mathbb{P}_{\mathrm{pc}}$.

As in the nondeterministic setting, the most convenient way to represent a test is through a process term, which interacts with any process term under test by means of a parallel composition operator that enforces synchronization on the set $Name_{\mathrm{v}}$ of all visible action names. Due to the adoption of an asymmetric synchronization discipline, a test can comprise only passive visible actions, so that the composite term inherits performance closure from the process term under test.

From a testing viewpoint, in any of its states a process term under test generates the proposal of an action to be executed by means of a race among the exponentially timed actions enabled in that state. If the name of the proposed action is τ, then the process term advances by itself. Otherwise, the test either reacts by participating in the interaction with the process term through a passive action having the same name as the proposed exponentially timed action, or blocks the interaction if it has no passive actions with the proposed name.

Markovian testing equivalence relies on comparing the process term probabilities of performing successful test-driven computations within arbitrary sequences of average amounts of time. Due to the presence of these average time upper bounds, for the test representation we can restrict ourselves to nonrecursive process terms. In other words, the expressiveness provided by finite-state labeled multitransition systems with an acyclic structure is enough for tests.

In order not to interfere with the quantitative aspects of the behavior of process terms under test, we avoid the introduction of a success action ω. The successful completion of a test is formalized in the text syntax by replacing $\underline{0}$ with a zeroary operator s denoting a success state. Ambiguous tests including several summands among which at least one equal to s are avoided through a two-level syntax.

Definition 3.1 The set $\mathbb{T}_{\mathrm{R}}$ of reactive tests is generated by the following syntax:

$$\boxed{\begin{array}{rcl} T & ::= & \mathrm{s} \mid T' \\ T' & ::= & <a, *_w>.T \mid T' + T' \end{array}}$$

where $a \in Name_{\mathrm{v}}$ and $w \in \mathbb{R}_{>0}$. ■

Definition 3.2 Let $P \in \mathbb{P}_{\mathrm{pc}}$ and $T \in \mathbb{T}_{\mathrm{R}}$. The interaction system of P and T is process term $P \,\|_{Name_{\mathrm{v}}}\, T \in \mathbb{P}_{\mathrm{pc}}$ and we say that:

- A configuration is a state of $[\![P \,\|_{Name_{\mathrm{v}}}\, T]\!]$, which is formed by a process and a test projection.
- A configuration is successful iff its test projection is s.
- A test-driven computation is a computation of $[\![P \,\|_{Name_{\mathrm{v}}}\, T]\!]$.
- A test-driven computation is successful iff it traverses a successful configuration.

We denote by $\mathscr{SC}(P,T)$ the multiset of successful computations of $P \,\|_{Name_{\mathrm{v}}}\, T$. ■

If a process term $P \in \mathbb{P}_{\mathrm{pc}}$ under test has no exponentially timed τ-actions as it was in [2], then for all reactive tests $T \in \mathbb{T}_{\mathrm{R}}$ it turns out that: (i) all the computations in $\mathscr{SC}(P,T)$ have a finite length due to the restrictions imposed on the test syntax; (ii) all the computations in $\mathscr{SC}(P,T)$ are independent of each other because of their maximality; (iii) the multiset $\mathscr{SC}(P,T)$ is finite because P and T are finitely branching. Thus, all definitions of Sect. 2.4 are applicable to $\mathscr{SC}(P,T)$ and also to $\mathscr{SC}_{\leq\theta}(P,T)$ for any sequence $\theta \in (\mathbb{R}_{>0})^*$ of average amounts of time.

In order to cope with the possible presence of exponentially timed τ-actions within P in such a way that all the properties above hold – especially independence – we have to consider subsets of $\mathscr{SC}_{\leq\theta}(P,T)$ including all successful test-driven computations of the same length. This is also necessary to distinguish among process terms comprising only exponentially timed τ-actions – like $<\tau,\lambda>.\underline{0}$ and $<\tau,\mu>.\underline{0}$, with $\lambda > \mu$, mentioned in Sect. 1 – as there is a single test, s, that those process terms can pass. The only option is to compare them after executing the same number of τ-actions.

Since no element of $\mathscr{SC}_{\leq\theta}(P,T)$ can be longer than $|\theta|$, we should consider every possible subset $\mathscr{SC}^{l}_{\leq\theta}(P,T)$ for $0 \leq l \leq |\theta|$. However, it is enough to consider $\mathscr{SC}^{|\theta|}_{\leq\theta}(P,T)$, as shorter successful test-driven computations can be taken into account when imposing prefixes of θ as average time upper bounds. Therefore, the novelty with respect to [2] is simply the presence of the additional constraint $|\theta|$.

Definition 3.3 Let $P_1, P_2 \in \mathbb{P}_{\mathrm{pc}}$. We say that P_1 is Markovian testing equivalent to P_2, written $P_1 \sim_{\mathrm{MT}} P_2$, iff for all reactive tests $T \in \mathbb{T}_{\mathrm{R}}$ and sequences $\theta \in (\mathbb{R}_{>0})^*$ of average amounts of time:

$$prob(\mathscr{SC}^{|\theta|}_{\leq\theta}(P_1,T)) = prob(\mathscr{SC}^{|\theta|}_{\leq\theta}(P_2,T))$$

■

Note that we have not defined a may equivalence and a must equivalence as in the nondeterministic case [11]. The reason is that in this Markovian framework the possibility and the necessity of passing a test are not sufficient to discriminate among process terms, as they are qualitative concepts. What we have considered here is a single quantitative notion given by the probability of passing a test (within an average time upper bound); hence, the definition of a single equivalence. This quantitative notion subsumes both the possibility of passing a test – which can be encoded as the probability of passing the test being greater than zero – and the necessity of passing a test – which can be encoded as the probability of passing the test being equal to one.

Although we could have defined Markovian testing equivalence as the kernel of a Markovian testing preorder, this has not been done. The reason is that such a preorder would have boiled down to an equivalence relation, because for each reactive test passed by P_1 within θ with a probability less than the probability with which P_2 passes the same test within θ, in general it is possible to find a dual reactive test for which the relation between the two probabilities is inverted.

Another important difference with respect to the nondeterministic case is that the presence of average time upper bounds makes it possible to decide whether a test is passed or not even if the process term under test can execute infinitely many exponentially timed τ-actions. In other words, τ-divergence does not need to be taken into account.

4 Basic Properties and Characterizations

First of all, we observe that, whenever exponentially timed τ-actions are absent, the new Markovian testing equivalence $\sim_{\mathrm{MT}}$ coincides with the old one defined in [2], which we denote by $\sim_{\mathrm{MT,old}}$. In the following, we use $\mathbb{P}_{\mathrm{pc,v}}$ to refer to the process terms of $\mathbb{P}_{\mathrm{pc}}$ that contain no exponentially timed τ-actions.

Proposition 4.1 Let $P_1, P_2 \in \mathbb{P}_{\mathrm{pc,v}}$. Then $P_1 \sim_{\mathrm{MT}} P_2 \iff P_1 \sim_{\mathrm{MT,old}} P_2$. ■

Then, we have two alternative characterizations of $\sim_{\mathrm{MT}}$, which provide further justifications for the way in which the equivalence has been defined. The first one establishes that the discriminating power does not change if we consider a set $\mathbb{T}_{\mathrm{R,lib}}$ of tests with the following more liberal syntax:

$$T \ ::= \ \mathrm{s} \mid <a, *_w>.T \mid T + T$$

provided that by successful configuration we mean a configuration whose test projection includes s as top-level summand. Let us denote by $\sim_{\mathrm{MT,lib}}$ the resulting variant of Markovian testing equivalence.

Proposition 4.2 Let $P_1, P_2 \in \mathbb{P}_{\mathrm{pc}}$. Then $P_1 \sim_{\mathrm{MT,lib}} P_2 \iff P_1 \sim_{\mathrm{MT}} P_2$. ■

The second characterization establishes that the discriminating power does not change if we consider a set $\mathbb{T}_{\mathrm{R},\tau}$ of tests capable of moving autonomously by executing exponentially timed τ-actions:

$$\begin{array}{rcl} T & ::= & \mathrm{s} \mid T' \\ T' & ::= & <a, *_w>.T \mid <\tau, \lambda>.T \mid T' + T' \end{array}$$

Let us denote by $\sim_{\mathrm{MT},\tau}$ the resulting variant of Markovian testing equivalence.

Proposition 4.3 Let $P_1, P_2 \in \mathbb{P}_{\mathrm{pc}}$. Then $P_1 \sim_{\mathrm{MT},\tau} P_2 \iff P_1 \sim_{\mathrm{MT}} P_2$. ■

Finally, we have two further alternative characterizations of $\sim_{\mathrm{MT}}$ coming from [2]. The first one establishes that the discriminating power does not change if we consider the (more accurate) probability distribution of passing tests within arbitrary sequences of amounts of time, rather than the (easier to work with) probability of passing tests within arbitrary sequences of average amounts of time.

The second characterization fully abstracts from comparing process term behavior in response to tests. This is achieved by considering traces that are extended at each step with the set of visible action

names permitted by the environment at that step (not to be confused with a ready set). A consequence of the structure of extended traces is the identification of a set $\mathbb{T}_{\mathrm{R,c}}$ of canonical reactive tests, which is generated by the following syntax:

$$\boxed{T \ ::= \ \mathrm{s} \mid <a,*_1>.T + \sum_{b \in \mathscr{E}-\{a\}} <b,*_1>.<\mathrm{z},*_1>.\mathrm{s}}$$

where $a \in \mathscr{E}$, $\mathscr{E} \subseteq Name_{\mathrm{v}}$ finite, the summation is absent whenever $\mathscr{E} = \{a\}$, and z is a visible action name representing failure that can occur within tests but not within process terms under test. Similar to the case of probabilistic testing equivalence [9, 10], each of these canonical reactive tests admits a single computation leading to success, whose intermediate states can have additional computations each leading to failure in one step. We point out that the canonical reactive tests are name deterministic, in the sense that the names of the passive actions occurring in any of their branches are all distinct.

5 Congruence Property

Markovian testing equivalence is a congruence with respect to all MPC operators. In particular, unlike [2], we have a full congruence result with respect to parallel composition.

Theorem 5.1 Let $P_1, P_2 \in \mathbb{P}_{\mathrm{pc}}$. Whenever $P_1 \sim_{\mathrm{MT}} P_2$, then:

1. $<a,\lambda>.P_1 \sim_{\mathrm{MT}} <a,\lambda>.P_2$ for all $<a,\lambda> \in Act$.
2. $P_1 + P \sim_{\mathrm{MT}} P_2 + P$ and $P + P_1 \sim_{\mathrm{MT}} P + P_2$ for all $P \in \mathbb{P}_{\mathrm{pc}}$.
3. $P_1 \,\|_S\, P \sim_{\mathrm{MT}} P_2 \,\|_S\, P$ and $P \,\|_S\, P_1 \sim_{\mathrm{MT}} P \,\|_S\, P_2$ for all $P \in \mathbb{P}$ and $S \subseteq Name_{\mathrm{v}}$ s.t. $P_1 \,\|_S\, P, P_2 \,\|_S\, P \in \mathbb{P}_{\mathrm{pc}}$.
4. $P_1/H \sim_{\mathrm{MT}} P_2/H$ for all $H \subseteq Name_{\mathrm{v}}$.
5. $P_1[\varphi] \sim_{\mathrm{MT}} P_2[\varphi]$ for all $\varphi \in Relab$. ■

It is worth stressing that the additional constraint on the length of successful test-driven computations present in Def. 3.3 is fundamental for achieving congruence with respect to alternative and parallel composition. As an example, if it were $<\tau,\lambda>.\underline{0} \sim_{\mathrm{MT}} <\tau,\mu>.\underline{0}$ for $\lambda > \mu$, then we would have $<\tau,\lambda>.\underline{0} + <a,\gamma>.\underline{0} \not\sim_{\mathrm{MT}} <\tau,\mu>.\underline{0} + <a,\gamma>.\underline{0}$. In fact, when the average time upper bound is high enough, the probability of passing $<a,*_1>.\mathrm{s}$ is $\frac{\gamma}{\lambda+\gamma}$ for the first term, whereas it is $\frac{\gamma}{\mu+\gamma}$ for the second term. We also mention that Props. 4.2 and 4.3 are exploited in the congruence proof for static operators.

6 Sound and Complete Axiomatization

Markovian testing equivalence has a sound and complete axiomatization over the set $\mathbb{P}_{\mathrm{pc,nrec}}$ of nonrecursive process terms of $\mathbb{P}_{\mathrm{pc}}$, given by the set $\mathscr{A}_{\mathrm{MT}}$ of equational laws of Table 2.

Apart from the usual laws for the alternative composition operator and for the unary static operators, unlike the axiomatization of [2] we now have laws dealing with concurrency. In particular, axiom $\mathscr{A}_{\mathrm{MT},5}$ concerning the parallel composition of $P \equiv \sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_i$ and $Q \equiv \sum_{j \in J} <b_j, \tilde{\mu}_j>.Q_j$ – where I and J are nonempty finite index sets and each summation on the right-hand side of the axiom is taken to be $\underline{0}$ whenever its set of summands is empty – is the expansion law when enforcing generative-reactive and reactive-reactive synchronizations. This axiom applies to non-performance-closed process terms too; e.g., the last addendum on its right-hand side is related to reactive-reactive synchronizations.

$$
\begin{array}{lrcl}
(\mathscr{A}_{\mathrm{MT},1}) & P_1 + P_2 & = & P_2 + P_1 \\
(\mathscr{A}_{\mathrm{MT},2}) & (P_1 + P_2) + P_3 & = & P_1 + (P_2 + P_3) \\
(\mathscr{A}_{\mathrm{MT},3}) & P + \underline{0} & = & P
\end{array}
$$

$$
(\mathscr{A}_{\mathrm{MT},4}) \quad \sum_{i \in I} <a, \lambda_i>. \sum_{j \in J_i} <b_{i,j}, \mu_{i,j}>.P_{i,j} \; = \; <a, \sum_{k \in I} \lambda_k>. \sum_{i \in I} \sum_{j \in J_i} <b_{i,j}, \frac{\lambda_i}{\sum_{k \in I} \lambda_k} \cdot \mu_{i,j}>.P_{i,j}
$$

if: I is a finite index set with $|I| \geq 2$;
for all $i \in I$, index set J_i is finite and its summation is $\underline{0}$ if $J_i = \emptyset$;
for all $i_1, i_2 \in I$ and $b \in \mathit{Name}$:

$$
\sum_{j \in J_{i_1}} \{\!|\, \mu_{i_1,j} \mid b_{i_1,j} = b \,|\!\} \; = \; \sum_{j \in J_{i_2}} \{\!|\, \mu_{i_2,j} \mid b_{i_2,j} = b \,|\!\}
$$

$$
\begin{aligned}
(\mathscr{A}_{\mathrm{MT},5}) \quad & \sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_i \parallel_S \sum_{j \in J} <b_j, \tilde{\mu}_j>.Q_j \; = \\
& \sum_{k \in I, a_k \notin S} <a_k, \tilde{\lambda}_k>. \left(P_k \parallel_S \sum_{j \in J} <b_j, \tilde{\mu}_j>.Q_j \right) + \\
& \sum_{h \in J, b_h \notin S} <b_h, \tilde{\mu}_h>. \left(\sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_i \parallel_S Q_h \right) + \\
& \sum_{k \in I, a_k \in S, \tilde{\lambda}_k \in \mathbb{R}_{>0}} \; \sum_{h \in J, b_h = a_k, \tilde{\mu}_h = *_{w_h}} <a_k, \tilde{\lambda}_k \cdot \frac{w_h}{\mathit{weight}(Q, b_h)}>.(P_k \parallel_S Q_h) + \\
& \sum_{h \in J, b_h \in S, \tilde{\mu}_h \in \mathbb{R}_{>0}} \; \sum_{k \in I, a_k = b_h, \tilde{\lambda}_k = *_{v_k}} <b_h, \tilde{\mu}_h \cdot \frac{v_k}{\mathit{weight}(P, a_k)}>.(P_k \parallel_S Q_h) + \\
& \sum_{k \in I, a_k \in S, \tilde{\lambda}_k = *_{v_k}} \; \sum_{h \in J, b_h = a_k, \tilde{\mu}_h = *_{w_h}} <a_k, *_{\mathit{norm}(v_k, w_h, a_k, P, Q)}>.(P_k \parallel_S Q_h)
\end{aligned}
$$

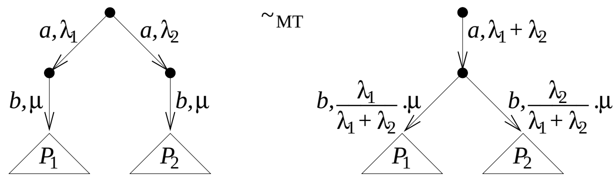
$$
\begin{array}{lrcl}
(\mathscr{A}_{\mathrm{MT},6}) & \sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_i \parallel_S \underline{0} & = & \sum_{k \in I, a_k \notin S} <a_k, \tilde{\lambda}_k>.P_k \\
(\mathscr{A}_{\mathrm{MT},7}) & \underline{0} \parallel_S \sum_{j \in J} <b_j, \tilde{\mu}_j>.Q_j & = & \sum_{h \in J, b_h \notin S} <b_h, \tilde{\mu}_h>.Q_h \\
(\mathscr{A}_{\mathrm{MT},8}) & \underline{0} \parallel_S \underline{0} & = & \underline{0}
\end{array}
$$

$$
\begin{array}{lrcll}
(\mathscr{A}_{\mathrm{MT},9}) & \underline{0}/H & = & \underline{0} & \\
(\mathscr{A}_{\mathrm{MT},10}) & (<a, \tilde{\lambda}>.P)/H & = & <\tau, \tilde{\lambda}>.(P/H) & \text{if } a \in H \\
(\mathscr{A}_{\mathrm{MT},11}) & (<a, \tilde{\lambda}>.P)/H & = & <a, \tilde{\lambda}>.(P/H) & \text{if } a \notin H \\
(\mathscr{A}_{\mathrm{MT},12}) & (P_1 + P_2)/H & = & P_1/H + P_2/H &
\end{array}
$$

$$
\begin{array}{lrcl}
(\mathscr{A}_{\mathrm{MT},13}) & \underline{0}[\varphi] & = & \underline{0} \\
(\mathscr{A}_{\mathrm{MT},14}) & (<a, \tilde{\lambda}>.P)[\varphi] & = & <\varphi(a), \tilde{\lambda}>.(P[\varphi]) \\
(\mathscr{A}_{\mathrm{MT},15}) & (P_1 + P_2)[\varphi] & = & P_1[\varphi] + P_2[\varphi]
\end{array}
$$

Table 2: Equational laws for $\sim_{\mathrm{MT}}$

Like in [2], the law characterizing $\sim_{\rm MT}$ is the axiom schema $\mathscr{A}_{\rm MT,4}$, which in turn subsumes the law $<a,\lambda_1>.P + <a,\lambda_2>.P = <a,\lambda_1+\lambda_2>.P$ characterizing Markovian bisimilarity. The simplest instance of axiom schema $\mathscr{A}_{\rm MT,4}$ is depicted below:



As emphasized by the figure above, $\sim_{\rm MT}$ allows choices to be deferred in the case of branches that start with the same action name (see the two a-branches on the left-hand side) and are followed by sets of actions having the same names and total rates (see $\{<b,\mu>\}$ after each of the two a-branches).

Theorem 6.1 Let $P_1, P_2 \in \mathbb{P}_{\rm pc,nrec}$. Then $\mathscr{A}_{\rm MT} \vdash P_1 = P_2 \iff P_1 \sim_{\rm MT} P_2$. ■

7 Modal Logic Characterization

Markovian testing equivalence has a modal logic characterization that, as in [4], is based on a modal language comprising true, disjunction, and diamond. A constraint is imposed on formulas of the form $\phi_1 \vee \phi_2$, which does not reduce the expressive power as it is consistent with the name-deterministic nature of branches within canonical reactive tests (see Sect. 4).

Definition 7.1 The set of formulas of the modal language $\mathscr{M}\mathscr{L}_{\rm MT}$ is generated by the following syntax:

$$\boxed{\begin{array}{lcl} \phi & ::= & \text{true} \mid \phi' \\ \phi' & ::= & \langle a \rangle \phi \mid \phi' \vee \phi' \end{array}}$$

where $a \in Name_{\rm v}$ and each formula of the form $\phi_1 \vee \phi_2$ satisfies:

$$init(\phi_1) \cap init(\phi_2) = \emptyset$$

with $init(\phi)$ being defined by induction on the syntactical structure of ϕ as follows:

$$\begin{array}{rcl} init(\text{true}) & = & \emptyset \\ init(\langle a \rangle \phi) & = & \{a\} \\ init(\phi_1 \vee \phi_2) & = & init(\phi_1) \cup init(\phi_2) \end{array}$$ ■

Probabilistic and temporal information do not decorate any operator of the modal language, but come into play through a quantitative interpretation function inspired by [16] that replaces the usual boolean satisfaction relation. This interpretation function measures the probability that a process term satisfies a formula quickly enough on average. The constraint imposed by Def. 7.1 on disjunctions guarantees that their subformulas exercise independent computations of the process term, thus ensuring the correct calculation of the probability of satisfying the overall formula. In order to manage exponentially timed τ-actions, unlike [4] the length of the computations satisfying the formula has to be taken into account as well.

Definition 7.2 The interpretation function $[\![.]\!]^{.}_{\rm MT}$ of $\mathscr{M}\mathscr{L}_{\rm MT}$ over $\mathbb{P}_{\rm pc} \times (\mathbb{R}_{>0})^*$ is defined by letting:

$$\boxed{[\![\phi]\!]^{|\theta|}_{\rm MT}(P,\theta) = \begin{cases} 0 & \text{if } |\theta| = 0 \wedge \phi \not\equiv \text{true or} \\ & \quad |\theta| > 0 \wedge rate_{\rm o}(P, init(\phi) \cup \{\tau\}, 0) = 0 \\ 1 & \text{if } |\theta| = 0 \wedge \phi \equiv \text{true} \end{cases}}$$

otherwise by induction on the syntactical structure of ϕ and on the length of θ as follows:

$$[\![\text{true}]\!]_{\text{MT}}^{|t\circ\theta|}(P, t\circ\theta) = \begin{cases} \sum\limits_{P \xrightarrow{\tau,\lambda} P'} \frac{\lambda}{rate_{\text{o}}(P,\tau,0)} \cdot [\![\text{true}]\!]_{\text{MT}}^{|\theta|}(P',\theta) & \text{if } \frac{1}{rate_{\text{o}}(P,\tau,0)} \leq t \\ 0 & \text{if } \frac{1}{rate_{\text{o}}(P,\tau,0)} > t \end{cases}$$

$$[\![\langle a\rangle\phi]\!]_{\text{MT}}^{|t\circ\theta|}(P, t\circ\theta) = \begin{cases} \sum\limits_{P \xrightarrow{a,\lambda} P'} \frac{\lambda}{rate_{\text{o}}(P,\{a,\tau\},0)} \cdot [\![\phi]\!]_{\text{MT}}^{|\theta|}(P',\theta) + & \\ \quad \sum\limits_{P \xrightarrow{\tau,\lambda} P'} \frac{\lambda}{rate_{\text{o}}(P,\{a,\tau\},0)} \cdot [\![\langle a\rangle\phi]\!]_{\text{MT}}^{|\theta|}(P',\theta) & \text{if } \frac{1}{rate_{\text{o}}(P,\{a,\tau\},0)} \leq t \\ 0 & \text{if } \frac{1}{rate_{\text{o}}(P,\{a,\tau\},0)} > t \end{cases}$$

$$\begin{aligned} [\![\phi_1 \vee \phi_2]\!]_{\text{MT}}^{|t\circ\theta|}(P, t\circ\theta) &= p_1 \cdot [\![\phi_1]\!]_{\text{MT}}^{|t_1\circ\theta|}(P_{\textit{no-init-}\tau}, t_1\circ\theta) + p_2 \cdot [\![\phi_2]\!]_{\text{MT}}^{|t_2\circ\theta|}(P_{\textit{no-init-}\tau}, t_2\circ\theta) \\ &+ \sum_{P \xrightarrow{\tau,\lambda} P'} \frac{\lambda}{rate_{\text{o}}(P,init(\phi_1\vee\phi_2)\cup\{\tau\},0)} \cdot [\![\phi_1 \vee \phi_2]\!]_{\text{MT}}^{|\theta|}(P',\theta) \end{aligned}$$

where $P_{\textit{no-init-}\tau}$ is P devoid of all of its computations starting with a τ-transition – which is assumed to be $\underline{0}$ whenever all the computations of P start with a τ-transition – and for $j \in \{1,2\}$:

$$p_j = \frac{rate_{\text{o}}(P,init(\phi_j),0)}{rate_{\text{o}}(P,init(\phi_1\vee\phi_2)\cup\{\tau\},0)} \qquad t_j = t + \left(\frac{1}{rate_{\text{o}}(P,init(\phi_j),0)} - \frac{1}{rate_{\text{o}}(P,init(\phi_1\vee\phi_2)\cup\{\tau\},0)}\right) \qquad \blacksquare$$

In the definition above, p_j represents the probability with which P performs actions whose name is in $init(\phi_j)$ rather than actions whose name is in $init(\phi_k) \cup \{\tau\}$, $k = 3 - j$, given that P can perform actions whose name is in $init(\phi_1 \vee \phi_2) \cup \{\tau\}$. These probabilities are used as weights for the correct account of the probabilities with which P satisfies only ϕ_1 or ϕ_2 in the context of the satisfaction of $\phi_1 \vee \phi_2$. If such weights were omitted, then the fact that $\phi_1 \vee \phi_2$ offers a set of initial actions at least as large as the ones offered by ϕ_1 alone and by ϕ_2 alone would be ignored, thus leading to a potential overestimate of the probability of satisfying $\phi_1 \vee \phi_2$.

Similarly, t_j represents the extra average time granted to P for satisfying only ϕ_j. This extra average time is equal to the difference between the average sojourn time in P when only actions whose name is in $init(\phi_j)$ are enabled and the average sojourn time in P when also actions whose name is in $init(\phi_k) \cup \{\tau\}$, $k = 3 - j$, are enabled. Since the latter cannot be greater than the former due to the race policy – more enabled actions means less time spent on average in a state – considering t instead of t_j in the satisfaction of ϕ_j in isolation would lead to a potential underestimate of the probability of satisfying $\phi_1 \vee \phi_2$ within the given average time upper bound, as P may satisfy $\phi_1 \vee \phi_2$ within $t \circ \theta$ even if P satisfies neither ϕ_1 nor ϕ_2 taken in isolation within $t \circ \theta$.

Theorem 7.3 $P_1 \sim_{\text{MT}} P_2 \iff \forall \phi \in \mathscr{ML}_{\text{MT}}. \forall \theta \in (\mathbb{R}_{>0})^*. [\![\phi]\!]_{\text{MT}}^{|\theta|}(P_1,\theta) = [\![\phi]\!]_{\text{MT}}^{|\theta|}(P_2,\theta)$. ■

8 Verification Algorithm

Markovian testing equivalence can be decided in polynomial time. The reason is that Markovian testing equivalence coincides with Markovian ready equivalence and, given two process terms, their underlying CTMCs in which action names have not been discarded from transition labels are Markovian ready equivalent iff the corresponding embedded DTMCs in which transitions have been labeled with suitably

augmented names are related by probabilistic ready equivalence. The latter equivalence is decidable in polynomial time [15] through a reworking of the algorithm for probabilistic language equivalence [18].

Following [19], the transformation of a name-labeled CTMC into the corresponding embedded name-labeled DTMC is carried out by simply turning the rate of each transition into the corresponding execution probability. Then, we need to encode the total exit rate of each state of the original name-labeled CTMC inside the names of all transitions departing from that state in the associated embedded DTMC.

Acknowledgment: This work has been funded by MIUR-PRIN project *PaCo – Performability-Aware Computing: Logics, Models, and Languages*.

References

[1] C. Baier, J.-P. Katoen, H. Hermanns, and V. Wolf, *"Comparative Branching-Time Semantics for Markov Chains"*, in Information and Computation 200:149-214, 2005.

[2] M. Bernardo, *"Non-Bisimulation-Based Markovian Behavioral Equivalences"*, in Journal of Logic and Algebraic Programming 72:3-49, 2007.

[3] M. Bernardo, *"Towards State Space Reduction Based on T-Lumpability-Consistent Relations"*, in Proc. of EPEW 2008, Springer, LNCS 5261:64-78, Palma de Mallorca (Spain), 2008.

[4] M. Bernardo, *"Uniform Logical Characterizations of Testing Equivalences for Nondeterministic, Probabilistic and Markovian Processes"*, in Proc. of QAPL 2009, Elsevier, ENTCS, York (UK), 2009.

[5] M. Bernardo and M. Bravetti, *"Performance Measure Sensitive Congruences for Markovian Process Algebras"*, in Theoretical Computer Science 290:117-160, 2003.

[6] M. Bravetti, *"Revisiting Interactive Markov Chains"*, in Proc. of MTCS 2002, Elsevier, ENTCS 68(5):1-20, Brno (Czech Republic), 2002.

[7] S.D. Brookes, C.A.R. Hoare, and A.W. Roscoe, *"A Theory of Communicating Sequential Processes"*, in Journal of the ACM 31:560-599, 1984.

[8] P. Buchholz, *"Exact and Ordinary Lumpability in Finite Markov Chains"*, in Journal of Applied Probability 31:59-75, 1994.

[9] I. Christoff, *"Testing Equivalences and Fully Abstract Models for Probabilistic Processes"*, in Proc. of CONCUR 1990, Springer, LNCS 458:126-140, Amsterdam (The Netherlands), 1990.

[10] R. Cleaveland, Z. Dayar, S.A. Smolka, and S. Yuen, *"Testing Preorders for Probabilistic Processes"*, in Information and Computation 154:93-148, 1999.

[11] R. De Nicola and M. Hennessy, *"Testing Equivalences for Processes"*, in Theoretical Computer Science 34:83-133, 1983.

[12] R.J. van Glabbeek, S.A. Smolka, and B. Steffen, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, in Information and Computation 121:59-80, 1995.

[13] H. Hermanns, *"Interactive Markov Chains"*, Springer, LNCS 2428, 2002.

[14] J. Hillston, *"A Compositional Approach to Performance Modelling"*, Cambridge University Press, 1996.

[15] D.T. Huynh and L. Tian, *"On Some Equivalence Relations for Probabilistic Processes"*, in Fundamenta Informaticae 17:211-234, 1992.

[16] M.Z. Kwiatkowska and G.J. Norman, *"A Testing Equivalence for Reactive Probabilistic Processes"*, in Proc. of EXPRESS 1998, Elsevier, ENTCS 16(2):114-132, Nice (France), 1998.

[17] R. Milner, *"Communication and Concurrency"*, Prentice Hall, 1989.

[18] W.-G. Tzeng, *"A Polynomial-Time Algorithm for the Equivalence of Probabilistic Automata"*, in SIAM Journal on Computing 21:216-227, 1992.

[19] V. Wolf, C. Baier, and M. Majster-Cederbaum, *"Trace Machines for Observing Continuous-Time Markov Chains"*, in Proc. of QAPL 2005, Elsevier, ENTCS 153(2):259-277, Edinburgh (UK), 2005.